

The applicant proposes the following claim amendments:

1 1. (Currently amended) In a system in which a hardware target computer
2 system, which has a target instruction set architecture (ISA), executes a target
3 instruction sequence corresponding to a source instruction sequence of a source
4 system, which has a source ISA and is running on the target computer system, a
5 method for handling exceptions comprising the following steps:
6 converting the source instruction sequence into the target instruction sequence
7 by binary translation, each instruction in the source instruction sequence being
8 converted into a corresponding translated target instruction sequence, which may
9 consist of a single target instruction;
10 determining beginning and ending addresses of each source instruction and
11 each corresponding translated target instruction;
12 generating a mapping between the beginning and ending addresses of each
13 source instruction and its corresponding translated target instruction sequence;
14 executing the translated target instruction sequence;
15 at any point in the translated target instruction sequence, sensing the presence
16 of an exception and determining whether each sensed exception is synchronous or
17 asynchronous,
18 a synchronous exception being defined as an exception resulting from attempted
19 execution of a target instruction, in which synchronous exceptions are of either of two
20 types, namely, transparent and non-transparent, a transparent exception being defined
21 as an exception requiring processing action wholly within the target computer system,
22 and a non-transparent exception being defined as an exception requiring processing
23 that alters a visible state of the source system; and
24 an asynchronous exception being defined as an exception resulting from an
25 event unrelated to the execution of a target instruction; and
26 if the sensed exception is asynchronous, resuming to completion the execution of
27 the target instruction sequence in binary translation from the point in the translated
28 target instruction sequence at which the asynchronous exception was sensed before
29 handling the asynchronous exception and thus delaying handling of each

BEST AVAILABLE COPY

30 asynchronous sensed exception until, and no later than, completion of execution of the
31 target instruction sequence corresponding to the current source instruction when the
32 asynchronous exception is sensed;
33 determining whether the sensed synchronous exception is transparent or non-
34 transparent;
35 handling each transparent synchronous exception externally from the source
36 system, the visible state of the source system thereby remaining unaltered; and
37 forwarding to the source system for processing each non-transparent
38 synchronous exception.

Claims 2-5.

1 6. (Currently amended) A method as in claim 5 1, in which the step of
2 forwarding each non-transparent synchronous exception to the source system includes
3 the step of converting the sensed exception into a simulated source exception in a
4 source instruction stream, which is sensed by and interrupts the source system.

7. (Canceled)

1 8. (Currently amended) A method as in claim 7 1, further including the
2 following steps:
3 determining a source instruction pointer as a predetermined function of the final
4 target instruction pointer;
5 forwarding and processing the sensed asynchronous exception;
6 resuming execution at the location in the translation cache that corresponds to a
7 current source instruction pointer; and
8 asynchronous exceptions thereby being processed only upon completion of
9 execution of the translated target instructions corresponding to whole source
10 instructions.

1 9. (Original) A method as in claim 8, in which the step of delaying processing
2 of the sensed asynchronous exception further includes the step of simulating execution
3 of the remaining target instructions.

1 10. (Original) A method as in claim 8, in which the step of delaying processing
2 of the sensed asynchronous exception further includes the step of single-stepping the
3 execution of the remaining target instructions.

1 11. (Original) A method as in claim 8, in which the step of delaying processing
2 of the sensed asynchronous exception further includes the following steps:
3 temporarily replacing with a trap generation instruction the initial target
4 instructions in each of the translated target instruction sequences that correspond to
5 target instruction sequences that possibly immediately follow the current target
6 instruction sequence;
7 resuming execution of the current target instruction sequence from the point at
8 which the asynchronous exception was sensed;
9 restoring each of the temporarily replaced instructions with their original content
10 after completion of the processing of the sensed asynchronous exception; and
11 upon reaching the trap generation instruction, forwarding and processing the
12 sensed asynchronous exception.

1 12. (Original) A method as in claim 8, in which the step of delaying the
2 processing of the sensed asynchronous exception further includes the following steps:
3 temporarily replacing with a trap generation instruction each indirect branch
4 instruction, each indirect branch instruction corresponding to a possible last instruction
5 of the current target instruction sequence;
6 resuming execution of the current target instruction sequence from the point at
7 which the asynchronous exception was processed;
8 restoring each of the temporarily replaced instructions with their original content;
9 and
10 simulating the restored indirect branch instruction.

1 13. (Original) A method as in claim 1, in which:
2 the source system is a virtual machine;
3 a virtual machine monitor that is operationally installed between the virtual
4 machine and the hardware target computer system, the virtual machine thereby running
5 on the virtual machine monitor;
6 the steps of converting the source instruction sequence into the target instruction
7 sequence by binary translation, executing the translated target instruction sequence,
8 sensing the presence of an exception, and delaying application of the sensed exception,
9 are carried out by the virtual machine monitor.

1 14. (Original) A method as in claim 1, in which the source ISA is identical to
2 the target ISA.

1 15. (Previously presented) In a system in which a hardware target computer
2 system, which has a target instruction set architecture (ISA), executes a target
3 instruction sequence corresponding to a source instruction sequence of a source
4 system, which has a source ISA and is running on the target computer system, a
5 method for handling exceptions comprising the following steps:

6 converting the source instruction sequence into the target instruction sequence
7 by binary translation, each instruction in the source instruction sequence being
8 converted into a corresponding translated target instruction sequence, which may
9 consist of a single target instruction;

10 executing the translated target instruction sequence;

11 at any point in the translated target instruction sequence, sensing the presence
12 of an exception,

13 each exception being of either of two types -- synchronous and
14 asynchronous -- a synchronous exception being defined as an exception resulting from
15 attempted execution of a target instruction and an asynchronous exception is defined as
16 an exception resulting from an event unrelated to the execution of a target instruction,

17 synchronous exceptions being of either of two types, namely, transparent
18 and non-transparent, a transparent exception being defined as an exception requiring
19 processing action wholly within the target computer system, and a non-transparent
20 exception being defined as an exception requiring processing that alters a visible state
21 of the source system;

22 determining whether each sensed exception is synchronous or asynchronous;

23 determining whether each sensed synchronous exception is transparent or non-
24 transparent;

25 upon sensing the presence of an asynchronous exception during execution of a
26 current one of the translated target instruction sequences, delaying processing of the
27 sensed asynchronous exception until completion of the remaining target instructions in
28 the current translated target instruction sequence;

29 determining beginning and ending addresses of each source instruction and
30 each corresponding translated target instruction;

31 generating a mapping between the beginning and ending addresses of each
32 source instruction and its corresponding translated target instruction sequence;

33 handling each transparent synchronous exception externally from the source
34 system, the visible state of the source system thereby remaining unaltered;
35 forwarding to the source system for processing each non-transparent
36 synchronous exception;
37 determining a source instruction pointer as a predetermined function of the final
38 target instruction pointer;
39 forwarding and processing each sensed asynchronous exception;
40 resuming execution at the location in the translation cache that corresponds to a
41 current source instruction pointer, that is, from the point in the translated target
42 instruction sequence at which the asynchronous exception was sensed, asynchronous
43 exceptions thereby being processed only upon completion of execution of the translated
44 target instructions corresponding to whole source instructions;
45 delaying handling of each asynchronous sensed exception until, and no later
46 than, completion of execution of the target instruction sequence corresponding to the
47 current source instruction when the asynchronous exception is sensed;
48 in which:
49 the source system is a virtual machine;
50 a virtual machine monitor that is operationally installed between the virtual
51 machine and the hardware target computer system, the virtual machine thereby running
52 on the virtual machine monitor, and
53 the steps of converting the source instruction sequence into the target instruction
54 sequence by binary translation, executing the translated target instruction sequence,
55 sensing the presence of an exception, and delaying application of the sensed exception,
56 are carried out by the virtual machine monitor.

1 16. (Currently amended) A system for virtualizing a computer system using
2 binary translation comprising:
3 a hardware target computer system that has a target instruction set architecture
4 (ISA) and executes a target instruction sequence;
5 a source system that has a source ISA and a source instruction sequence;
6 an intermediate software layer forming an interface between the source system
7 and the hardware target computer system;
8 a binary translator included within the intermediate software layer and converting
9 the source instruction sequence into the target instruction sequence by binary
10 translation, each instruction in the source instruction sequence being converted into a
11 corresponding translated target instruction sequence, which may consist of a single
12 target instruction;
13 an exception handler included within the intermediate software layer and, at any
14 point in the translated target instruction sequence, sensing the presence of an exception
15 and determining whether the exception is synchronous or asynchronous, in which
16 an asynchronous exception is defined as an exception resulting from an event
17 unrelated to the execution of a target instruction;
18 a synchronous exception is defined as an exception resulting from attempted
19 execution of a target instruction and is of either of two types, namely, transparent and
20 non-transparent, a transparent exception being defined as an exception requiring
21 processing action wholly within the target computer system, and a non-transparent
22 exception being defined as an exception requiring processing that alters a visible state
23 of the source system;
24 a resume component within the intermediate software layer for resuming to
25 completion the execution of the target instruction sequence in binary translation from
26 the point in the translated target instruction sequence at which the asynchronous
27 exception was sensed if the sensed exception is asynchronous, before handling of the
28 asynchronous exception and
29 the exception handler further delaying handling of each asynchronous sensed
30 exception until, and no later than, completion of execution of the target instruction

31 sequence corresponding to the current source instruction when the asynchronous
32 exception is sensed;
33 in which the exception handler is further provided:
34 for determining whether the sensed synchronous exception is transparent
35 or non-transparent;
36 for handling each transparent synchronous exception externally from the
37 source system, the visible state of the source system thereby remaining unaltered; and
38 for forwarding to the source system for processing each non-transparent
39 synchronous exception.

1 17. (Previously presented) A system as in claim 16, further including a map
2 containing a mapping between beginning and ending addresses of each source
3 instruction and its corresponding translated target instruction sequence.

18. (Canceled)

19. (Canceled)

1 20. (Currently amended) A system as in claim 49 16, in which the exception
2 handler is further provided for converting the sensed exception into a simulated source
3 exception in a source instruction stream, which is sensed by and interrupts the source
4 system.

1 21. (Currently amended) A system as in claim 49 16, in which the exception
2 handler is further provided, upon sensing the presence of an asynchronous exception
3 during execution of a current one of the translated target instruction sequences, for
4 delaying processing of the sensed asynchronous exception until completion of the
5 remaining target instructions in the current translated target instruction sequence.

1 22. (Currently amended) A system as in claim 16, in which the exception
2 handler is further provided
3 for determining a source instruction pointer as a predetermined function of the
4 final target instruction pointer;
5 for forwarding and processing the sensed asynchronous exception; and
6 for resuming execution at the location in the translation cache that corresponds
7 to a current source instruction pointer, asynchronous exceptions thereby being
8 processed only upon completion of execution of the translated target instructions
9 corresponding to whole source instructions.

1 23. (Previously presented) A system as in claim 16, in which:
2 the source system is a virtual machine;
3 the system further includes a virtual machine monitor that is operationally
4 installed between the virtual machine and the hardware target computer system, the
5 virtual machine thereby running on the virtual machine monitor; and
6 the binary translator and exception handler are included within the virtual
7 machine monitor.

1 24. (Original) A method as in claim 16, in which the source ISA is identical to
2 the target ISA.

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☐ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKewed/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☒ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER: _____**

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.